

Evaluating Simulated Annealing Algorithms in the Optimization of Bacterial Strains

Miguel Rocha¹, Rui Mendes¹, Paulo Maia¹, José P. Pinto¹, Isabel Rocha²,
and Eugénio C. Ferreira²

¹ Departament of Informatics / CCTC - University of Minho
Campus de Gualtar, 4710-057 Braga - Portugal

mrocha@di.uminho.pt, rcm@di.uminho.pt, paulo.maia@di.uminho.pt

² IBB - Institute for Biotechnology and Bioengineering
Center of Biological Engineering - University of Minho
Campus de Gualtar, 4710-057 Braga - Portugal
irocha@deb.uminho.pt, ecferreira@deb.uminho.pt

Abstract. In this work, a Simulated Annealing (SA) algorithm is proposed for a Metabolic Engineering task: the optimization of the set of gene deletions to apply to a microbial strain to achieve a desired production goal. Each mutant strain is evaluated by simulating its phenotype using the Flux-Balance Analysis approach, under the premise that microorganisms have maximized their growth along natural evolution. A set based representation is used in the SA to encode variable sized solutions, enabling the automatic discovery of the ideal number of gene deletions. The approach was compared to the use of Evolutionary Algorithms (EAs) to solve the same task. Two case studies are presented considering the production of succinic and lactic acid as the target, with the bacterium *E. coli*. The variable sized SA seems to be the best alternative, outperforming the EAs, showing a fast convergence and low variability among the several runs and also enabling the automatic discovery of the ideal number of knockouts.

Keywords: Simulated Annealing, Set based representations, Variable size chromosomes, Metabolic Engineering, Flux-Balance Analysis.

1 Introduction

Metabolic Engineering has been generating tools appropriate to introduce directed genetic modifications in microorganisms, to make them fit to comply with industrial purposes, i.e. to be able to synthesize some desired compounds, rather than to follow their natural aims (e.g. the maximization of growth) [15][10]. The importance of those approaches has been increasing as many traditional chemical processes are being replaced by biotechnology for the production of valuable products, such as pharmaceuticals, fuels or food ingredients.

Most often, these processes imply that the microorganism's metabolism needs to be modified, a task that can be complex. Current methods are still based mostly on intuitive design principles and scarcely on effective mathematical

models that can predict cellular behaviour. Nevertheless, and although whole cell models are still far away, it is possible to predict cellular metabolism under some simplifying assumptions, using existing mathematical models of metabolism.

One of the most important approaches in this direction considers the cell to be in a steady-state, i.e., the concentrations of all the metabolites is considered constant throughout time. This imposes a number of constraints over the fluxes of all reactions, that can be used to predict cellular behavior. This is the basis of the Flux Balance Analysis approach [7], where a particular flux is typically optimized using linear programming. In the most usual case, a flux for biomass production is defined, whose maximization is taken as the objective function, thus assuming that the microbes have evolved towards optimal growth [6]. Solving this optimization problem for genome-scale models results in getting the values for all the fluxes of the reactions occurring in the cell.

In this way, it is possible to predict the behavior of a microorganism, both in its wild type and also in mutant forms. This allows the definition of a bi-level optimization problem, adding a layer that searches for the best mutant that can be obtained from the wild type by applying a selected set of genetic modifications. In this work, this set will be restricted to the possibility of deleting genes from the wild type. The idea is to force the microorganisms to synthesize a desired product by selected gene deletions. Therefore, the underlying optimization problem consists in reaching an optimal subset of gene deletions to optimize an objective function related with the production of a given compound.

A first approach to this problem was proposed by the *OptKnock* algorithm [2], where mixed integer linear programming methods are used to reach a guaranteed optimum solution. This algorithm suffered from two important drawbacks: the impossibility of considering nonlinear objective functions and the considerable computation time required that only allowed the problem to be solved for a relatively small number of reactions.

An alternative approach was proposed by the *OptGene* algorithm [11] that considers the application of Evolutionary Algorithms (EAs). EAs are capable of providing near optimal solutions in a reasonable amount of time and also allow the optimization of nonlinear objective functions. *OptGene* proposes EAs with two alternative representation schemes: binary or integer. The first is closer to the natural evolution, but is more complex and leads to solutions with a large number of knockouts. The latter allowed for a more compact encoding scheme, representing only the gene deletions. One of its major limitations is the need to define *a priori* the number of gene knockouts, that remains fixed. In [13] an EA with a set-based representation is proposed to extend *OptGene*. The use of variable-sized chromosomes to encode the sets was a major improvement, allowing the automatic definition of the ideal number of gene deletions.

In this work, an alternative optimization strategy is proposed based on the use of a Simulated Annealing (SA) algorithm. The proposed algorithm encodes solutions using a variable size set-based representation, making use of mutation operators similar to the ones used by the EAs. The proposed algorithm will be

tested, using as a benchmark the two case studies proposed in [13] and comparing the results obtained with the ones achieved by the EA's.

2 Simulation Algorithms for the Prediction of Metabolic Behavior

One of the many potential applications of the recently sequenced and annotated genomes of microorganisms is the reconstruction of genome-scale metabolic networks. The set of metabolic reactions obtained can therefore be used to simulate the phenotypic behaviour of microorganisms. One approach is to write dynamic mass balances for each metabolite in the network, generating a set of ordinary differential equations that may be used to simulate the dynamic behavior of metabolite concentrations. However, there is still insufficient data on kinetic expressions and parameters, and it is only possible to simulate dynamic conditions for a few pathways[3].

Therefore, a steady state approximation is generally applied, where for each metabolite in the network, the sum of all productions and consumptions will be zero, weighted by the stoichiometric coefficients. Thus, for metabolite i , where $i = 1, \dots, M$ (M is the number of metabolites) the following constraint is defined:

$$\sum_{j=1}^N S_{ij}v_j = 0 \quad (1)$$

where S_{ij} is the stoichiometric coefficient for metabolite i in reaction j and v_j is the reaction rate or flux over the reaction j . It is possible to define a matrix S , composed of the S_{ij} values, $j = 1, \dots, N$ (N is the number of reactions); v is the N -dimensional vector of the fluxes of the reactions.

The mass balances are therefore reduced to a set of linear homogeneous equations. The maximum/minimum values of the fluxes can be set by additional constraints in the form $\alpha_j \leq v_j \leq \beta_j$, that are also used to specify both thermodynamic and environmental conditions (e.g. availability of nutrients).

For most of the metabolic networks, and because the number of fluxes is greater than the number of metabolites, the set of linear equations obtained from the application of Equation 1 to the M metabolites usually leads to an under-determined system, for which there exists an infinite number of feasible flux distributions that satisfy the constraints. However, if a given linear function over the fluxes is chosen to be maximized, it is possible to obtain a single solution by applying standard algorithms (e.g. *simplex*) for linear programming problems. This methodology is known as Flux Balance Analysis (FBA) [7].

The combination of this technique with the existence of validated genome-scale stoichiometric models [4][1] allows to simulate the phenotypic behaviour of a microorganism under defined environmental conditions without performing any experiments. The most common flux chosen for maximization is the biomass, based on the premise that microorganisms have maximized their growth along natural evolution, a premise that has been confirmed experimentally in some cases [6].

3 Simulated Annealing

Simulated Annealing (SA) is an optimization algorithm inspired in the annealing process used in metallurgy, where a melt, initially at high temperature, is slowly cooled so that the system at any time is approximately in thermodynamic equilibrium. As the cooling proceeds, the system becomes more ordered, approaching a minimal energy state when the temperature reaches zero. If the initial temperature of the system is too low or the cooling process is not sufficiently slow the system may become trapped in a local minimum energy state.

In the original Metropolis scheme, an initial state of a thermodynamic system is chosen at energy E and holding temperature T constant. The initial configuration is perturbed and the change in energy ΔE is computed. A better configuration is always accepted, while a worse configuration is only accepted with a probability given by the Boltzmann factor

$$p[\text{accept}] = e^{-\frac{\Delta E}{T}} \quad (2)$$

This process is then repeated a number of trials, sufficient to give good sampling statistics for the current temperature, and then the temperature is decreased according to a given cooling schedule. The entire process is repeated until the temperature is sufficiently low.

This Monte Carlo approach may be used to optimize real or combinatorial problems [8]. The current state is a solution to the optimization problem and the energy represents its objective function value. The solution is perturbed by any process that will generate a new solution from the current one, also denoted as a *mutation* operator. This perturbation may depend on the temperature, thus allowing larger steps to be taken when the temperature is high and fine-tuning when the temperature is low. The implementation used in this work allows the use of any kind of mutation or combination thereof to perturb the current solution and generate a new one. Any number of mutations may be applied, each with a given probability (that must sum 1). The solution encoding is also very flexible and may use for instance a binary representation, real values, sets, permutations, trees, etc.

The configuration parameters for the algorithm are the initial and final temperatures, the number of iterations performed at each temperature and the cooling schedule used. The choice of these parameters is of paramount importance to the performance of the algorithm. If the initial temperature is too low or the cooling schedule is not slow enough, the optimization process may become stuck in a local optimum. On the other hand, if the initial temperature is too high, the cooling is too slow or the number of iterations per temperature is too high, the algorithm wastes a potentially large amount of computational time while searching for solutions. The cooling schedule used in this paper is among the most popular ones, where the temperature decrease is exponential, defined according to the following equation:

$$T_{n+1} = \alpha T_n \quad (3)$$

where $0 < \alpha \leq 1$. To ensure that the cooling schedule is sufficiently slow, the parameter α should be given values close to the unity.

As the choice of initial (T_0) and final temperatures (T_f) is problem dependent, it was decided to use the following configuration parameters:

ΔE_0 – The difference in energy that corresponds to an acceptance probability of 50% of worse solutions at the beginning of the run;

ΔE_f – The difference in energy that corresponds to an acceptance probability of 50% of worse solutions at the end of the run;

trials – The number of iterations per temperature;

NFEs – The number of function evaluations.

Using these parameters, the initial temperature, the final temperature and the scale parameter were computed using the following equations

$$T_0 = -\frac{\Delta E_0}{\log 0.5} \quad (4)$$

$$T_f = -\frac{\Delta E_f}{\log 0.5} \quad (5)$$

$$\alpha = \exp \left(\frac{\log T_f - \log T_0}{\left\lfloor \frac{\text{NFEs}}{\text{trials}} \right\rfloor} \right) \quad (6)$$

The advantage of using ΔE_0 and ΔE_f is that it allows the user who knows the fitness landscape of the optimization problem to automatically define the temperatures by reasoning over the values of the objective function. Supplying the number of function evaluations instead of the scale parameter α allows the user to accurately define the number of function evaluations the optimization algorithm will use, enabling a simpler comparison with other approaches.

4 The Proposed Algorithm

4.1 Representation Scheme and Mutation Operator

The problem addressed in this work consists in selecting, from a set of genes in a microbe's genome, a subset to be deleted in order to maximize a given objective function related to the microorganism's metabolism. The encoding of a solution is achieved by a set-based representation, where only gene deletions are represented. Each solution consists of a set of integer values representing the genes that will be deleted. Therefore, if the value i is in the set, this means the i -th gene in the microbe's genome is knocked out. Each value in the set is an integer with a value between 1 and N .

Two variants of this representation can be defined, considering fixed or variable sized sets. In the fixed-size alternative, the mutation operator creates solutions always of the same size. A random mutation operator is used that replaces a gene by a random value in the allowed range, avoiding duplicates in the set. In

variable-sized representations, sets with distinct cardinalities can be encoded and compete in the search process. In this case, two additional mutation operators are defined to be able to create solutions with a distinct size:

- *Grow*: consists in the introduction of a new gene into the chromosome, whose value is randomly generated in the available range (avoiding duplicates in the set).
- *Shrink*: a randomly selected gene is removed from the genome.

In the SA, the *Grow* and *Shrink* mutations are each used with a probability of 25% each, meaning that half of the new individuals are created in this way. The remaining are created by the aforementioned random mutation operator. In the experiments reported in this work, when a variable size is used, the minimum size is set to 1 and the maximum size is set to the number of genes (N), thus not restricting the possible range of solutions.

4.2 Decoding and Evaluating

The principle considered is a correspondence between the values in the set and metabolic reactions, i.e., each value represented in the set represents a particular enzyme that catalyzes a metabolic reaction. That enzyme is associated with a particular gene (or genes) that should be deleted for that reaction to be eliminated. The decoding process works by taking each value in the set and forcing the flux it indexes to the value 0, therefore disabling that reaction from the metabolic model. The process proceeds with the simulation of the mutant using FBA. The output is the set of values for the fluxes of all reactions, that are then used to compute the fitness value, given by an appropriate objective function.

One possible objective function is the Biomass-Product Coupled Yield (BPCY) [11], given by:

$$BPCY = \frac{PG}{S} \quad (7)$$

where P stands for the flux representing the excreted product; G for the organism's growth rate (biomass flux) and S for the substrate intake flux. Besides optimizing for the production of the desired product, this function also allows to select for mutants that exhibit high growth rates, i.e., that are likely to exhibit a higher productivity, an important industrial aim.

An alternative is to maximize only the value of the product's flux (P), but imposing a minimum threshold to the value of the biomass (G_{min}). Therefore, the objective function (denoted as Product Flux with Minimum Biomass (PFMB)) will be defined as: $PFMB = P$, if $G > G_{min}$; otherwise $PFMB = 0$.

4.3 Initialization

The initial solution is a set with randomly generated elements. In the variable size variant, the size of the individual is randomly created in the range [1,12]. The same process is used in the EAs to initialize each individual in the population.

4.4 Pre-processing and Post-processing

In genome-scale models the number of variables (fluxes over metabolic reactions) is in the order of hundreds or a few thousands and therefore the search space is very hard to address. Thus, every operation that gives a contribution to reduce this number, greatly improves the convergence of the algorithms. In this work, a number of operations is implemented to reduce the search space:

- Removal of fluxes that, given the constraints of the linear programming problem, cannot exhibit values different from 0.
- Equivalent variables, i.e. pairs of variables that are constrained to have the same value by the model. Each group of equivalent variables is replaced by a single variable.
- Discovery of essential genes that can not be deleted from the microorganism genome. As these genes should not be considered as targets for deletion, the search space for optimization is reduced. This list can be manually edited to include genes that are known to be essential, although that information can not be reached from the mathematical model.
- Identification of artificial fluxes that are associated with external metabolites and exchange fluxes that represent transport reactions. These are not allowed to be knocked out, since generally this would not have a biological meaning.

The best solution in each run goes through a simplification process, by identifying all gene deletions that contribute to the fitness of the solution, removing all deletions that keep the objective function unaltered. The aim is to keep only the necessary knockouts, given that the practical implementation of a gene deletion is both time consuming and costly.

4.5 Implementation Issues

The implementation of the proposed algorithms was performed by the authors in the *Java* programming language. In the implementation of FBA, the *GNU linear programming package (GLPK)*¹ was used to run the *simplex* algorithm.

5 Experiments

5.1 Experimental Setup

Two case studies were used to test the aforementioned algorithms. Both consider the microorganism *Escherichia coli* and the aim is to produce succinic and lactic acid (case studies I and II, respectively), with glucose as the limiting substrate. The genome-scale model for this microorganism used in the simulations was developed by Reed et al [12]. This model considers the *E. coli* metabolic network, including a total of $N = 1075$ fluxes and $M = 761$ metabolites. After the pre-processing stages, the simplified model remains with $N = 550$ and $M = 332$

¹ <http://www.gnu.org/software/glpk/>

metabolites. Furthermore, 227 essential genes are identified, which leaves 323 variables to be considered by the optimization algorithms.

The proposed SA is compared to the EAs proposed in [13]. Both algorithms were implemented in its fixed and variable size versions. In the first case, the cardinality of the set (k) took a number of distinct values. In the EA the population size was set to 100. The SA used $\Delta E_0 = 0.005$, $\Delta E_f = 5E - 5$ and $trials = 50$. In both cases, the termination criteria was defined based on a maximum of 50000 fitness evaluations. For each experimental setup, the process was repeated for 30 runs and the mean and 95% confidence intervals were calculated.

5.2 Case Study I: Succinic Acid

Succinic acid is one of the key intermediates in cellular metabolism and therefore an important case study for metabolic engineering[9]. The knockout solutions that lead to an improved phenotype regarding its production are not straightforward to identify since they involve a large number of interacting reactions. Succinic acid and its derivatives have been used as common chemicals to synthesize polymers, as additives and flavoring agents in foods, supplements for pharmaceuticals, or surfactants. Currently, it is produced through petrochemical processes that can be expensive and have significant environmental impacts.

In Table 1, the results for the EAs and SA, both fixed (the number of knockouts k is given) and variable sized (last row) are given, taking the BPCY as the objective function. The results show the mean, the 95% confidence interval and the maximum value of the BPCY for each configuration. In the last column, the mean of the number of gene deletions (after the simplification process) is shown.

In Table 1 it is possible to observe that, when using set-based representations with fixed size chromosomes, the results improve with the increase on the number of gene deletions. The improvement obtained when increasing from 6 to 20 gene deletions is essentially visible in the increase of the mean, since the best solution suffers minor improvements. Furthermore, there is less variability in the results, given by the smaller confidence intervals. The variable size alternatives seem to be able to automatically find the appropriate number of gene deletions. They also present a very low variability, given the small confidence interval.

Table 1. Results obtained for the case study I - production of succinic acid

k	EA				SA			
	Mean	Conf. int.	Best	Knockouts	Mean	Conf. int.	Best	Knockouts
2	0.0458	± 0.0288	0.0752	2.0	0.0475	± 0.0290	0.0752	2.0
4	0.1172	± 0.0769	0.3366	3.6	0.1096	± 0.0674	0.3440	3.7
6	0.2458	± 0.1108	0.3573	5.8	0.3184	± 0.1121	0.3573	5.9
8	0.2963	± 0.0969	0.3577	7.1	0.3401	± 0.0781	0.3576	7.2
10	0.3218	± 0.0739	0.3578	8.1	0.3566	± 0.0554	0.3578	8.1
12	0.3496	± 0.0370	0.3578	8.8	0.3573	± 0.0015	0.3578	8.7
20	0.3575	± 0.0012	0.3578	10.8	0.3577	± 0.0001	0.3578	10.2
VS	0.3507	± 0.0372	0.3579	11.8	0.3577	± 0.0001	0.3579	11.4

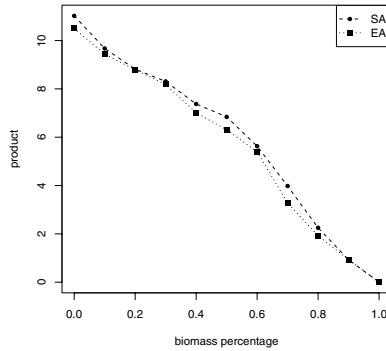


Fig. 1. Convergence plots for the variable-sized EA and SA in case study I

A comparison between EAs and SA show that the latter seems to obtain better results. When k is small (2 or 4) the results are comparable; when k increases the mean of the SA improves faster. Although some of the confidence intervals are overlapping, the standard deviations are smaller in the SA showing less variability in the results. Regarding the variable size versions, the SA has a better mean and a very reduced confidence interval, denoting the capacity to consistently find high quality results.

A distinct set of results was obtained by running both algorithms (variable sized variants) with the PYMB as the objective function. The minimum biomass was varied as a percentage of the wild type's value, in 10% intervals. The mean of the results for each algorithm is plotted in Figure 1. It is noticeable that the SA has slightly higher mean values for almost every point in the graph although the confidence intervals are overlapping in most cases (these are not shown for improved visualization). Nevertheless, the fact that SA achieves higher means and lower standard deviation values is an additional indicator of its performance.

5.3 Case Study II - Lactic Acid

Lactic acid and its derivatives have been used in a wide range of food-processing and industrial applications like meat preservation, cosmetics, oral and health care products and baked goods. Additionally, and because lactate can be easily converted to readily biodegradable polyesters, it is emerging as a potential material for producing environmentally friendly plastics from sugars [5]. Several microorganisms have been used to produce lactic acid, such as *Lactobacillus* strains. However, those bacteria have undesirable traits, such as a requirement for complex nutrients which complicates acid recovery. *E. coli* has many advantageous characteristics, such as rapid growth and simple nutritional requirements.

In Table 2, the results for the case study II are given. The first conclusion that can be drawn is that this case study seems to be less challenging than the one presented on the previous section. In fact, 3 gene deletions seem to be enough to obtain the best solution, and therefore the results with k larger than 6 are not

Table 2. Results obtained by for the case study II - production of lactic acid

k	EA				SA			
	Mean	Conf. int.	Best	Knockouts	Mean	Conf. int.	Best	Knockouts
2	0.2547	±0.0000	0.2547	2.0	0.2547	±0.0000	0.2547	2.0
4	0.2553	±0.0000	0.2553	3.0	0.2553	±0.0000	0.2553	3.0
6	0.2553	±0.0000	0.2553	3.0	0.2553	±0.0000	0.2553	3.0
VS	0.2553	±0.0000	0.2553	3.0	0.2553	±0.0000	0.2553	3.0

shown. On the other hand, the variable size algorithms confirm its merits once they are again able to find the best solution in all runs, and automatically finds the adequate number of knockouts.

5.4 Discussion

Two features that are important when comparing meta-heuristic optimization algorithms are the computational effort required and the convergence of the algorithm to a good solution. The computational burden of the alternatives compared is approximately the same, since the major computational effort is devoted to fitness evaluation and the same number of solutions is evaluated in every case. A typical run of each algorithm for the case studies presented will take approximately two or three hours in a regular PC.

Regarding the convergence of the algorithms, a plot of the evolution of the objective function along the generations of the SA and EA is given in Figure 2 (the mean of the 30 runs is plotted). Only the variable sized versions were selected to allow a better visualization. It is clear from this plot that the SA converges faster than the EA, obtaining high quality results early in the runs. Both

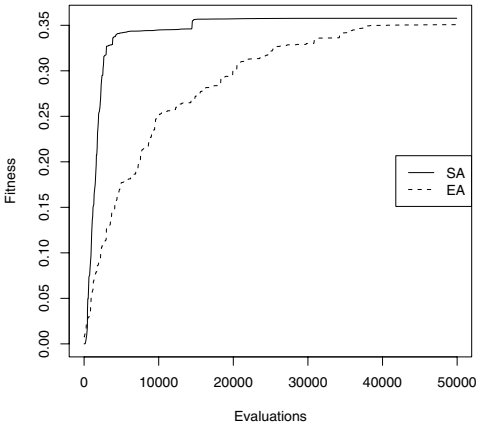


Fig. 2. Convergence plots for the variable-sized EA and SA in case study I

algorithms are similar in terms of computational effort, since most of the time is spent evaluating the solutions.

It is also important to refer that the approach followed is to solve the problem *in silico*, using computer models. Therefore, the results may or may not be biologically feasible. Nevertheless, a validation of the results was conducted since the best solutions obtained were analyzed by experts in Biotechnology using Bioinformatics databases (e.g. *EcoCyc*²).

6 Conclusions and Further Work

In this work, a contribution to Metabolic Engineering was provided by the development of a variant of Simulated Annealing that is able of reaching a near optimal set of gene deletions in a microbial strain, to maximize the production of a given product. This algorithm was able to improve previous results from the use of Evolutionary Algorithms. These were tested in a case study that dealt with the production of succinic acid by the *E. coli* bacterium. Important contributions of this work were the introduction of a set-based representation, that made use of variable size solutions, an uncommon feature in SA algorithms.

There are still a number of features that need to be introduced. These include other algorithms for simulation and distinct objective functions. Regarding the former, an alternative algorithm for simulating mutants' phenotype is the MOMA algorithm, that was proposed by Segre et al [14], where it is assumed that knockout metabolic fluxes undergo a minimal redistribution with respect to the flux configuration of the wild type. This implies solving a quadratic programming problem, whose aim is to minimize the differences between the fluxes in the mutant and the ones in the wild type. It would also be interesting to consider an objective function capable of taking into account the number of knockouts of a given solution and the cost of its experimental implementation.

Acknowledgments

The authors thank the Portuguese Foundation for Science and Technology (FCT) for their support through project ref. POSC/EIA/59899/2004. partially funded by FEDER.

References

1. Borodina, I., Nielsen, J.: From genomes to in silico cells via metabolic networks. *Current Opinion in Biotechnology* 16(3), 350–355 (2005)
2. Burgard, A.P., Pharya, P., Maranas, C.D.: Optknock: A bilevel programming framework for identifying gene knockout strategies for microbial strain optimization. *Biotechnol. Bioeng.* 84, 647–657 (2003)

² <http://www.ecocyc.org>

3. Chassagnole, C., Noisommit-Rizzi, N., Schmid, J.W., Mauch, K., Reuss, M.: Dynamic modeling of the central carbon metabolism of *escherichia coli*. *Biotechnology and Bioengineering* 79(1), 53–73 (2002)
4. Covert, M.W., Schilling, C.H., Famili, I., Edwards, J.S., Goryanin, I.I., Selkov, E., Palsson, B.O.: Metabolic modeling of microbial strains in silico. *Trends in Biochemical Sciences* 26(3), 179–186 (2001)
5. Hofvendahl, K., Hahn-Hagerdal, B.: Factors affecting the fermentative lactic acid production from renewable resources. *Enzyme Microbial Technology* 26, 87–107 (2000)
6. Ibarra, R.U., Edwards, J.S., Palsson, B.G.: *Escherichia coli* k-12 undergoes adaptive evolution to achieve in silico predicted optimal growth. *Nature* 420, 186–189 (2002)
7. Kauffman, K.J., Prakash, P., Edwards, J.S.: Advances in flux balance analysis. *Curr. Opin. Biotechnol.* 14, 491–496 (2003)
8. Kirkpatrick, S., Gelatt Jr., C.D., Vecchi, M.P.: Optimization by simulated annealing. *Science* 220(4598), 671–680 (1983)
9. Lee, S.Y., Hong, S.H., Moon, S.Y.: In silico metabolic pathway analysis and design: succinic acid production by metabolically engineered *escherichia coli* as an example. *Genome Informatics* 13, 214–223 (2002)
10. Nielsen, J.: Metabolic engineering. *Appl. Microbiol. Biotechnol.* 55, 263–283 (2001)
11. Patil, K., Rocha, I., Forster, J., Nielsen, J.: Evolutionary programming as a platform for in silico metabolic engineering. *BMC Bioinformatics* 6(308) (2005)
12. Reed, J.L., Vo, T.D., Schilling, C.H., Palsson, B.O.: An expanded genome-scale model of *escherichia coli* k-12 (ijr904 gsm/gpr). *Genome Biology* 4(9) R54.1–R54.12 (2003)
13. Rocha, M., Pinto, J.P., Rocha, I., Ferreira, E.C.: Optimization of Bacterial Strains with Variable-Sized Evolutionary Algorithms. In: *Proceedings of the IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology*, pp. 331–337. IEEE Press, Honolulu, USA (2007)
14. Segre, D., Vitkup, D., Church, G.M.: Analysis of optimality in natural and perturbed metabolic networks. *Proc. National Acad. Sciences USA* 99, 15112–15117 (2002)
15. Stephanopoulos, G., Aristidou, A.A., Nielsen, J.: *Metabolic engineering principles and methodologies*. Academic Press, San Diego (1998)